600102

NORMALIZED CUT ALGORITHM FOR AUTOMATED ASSIGNMENT OF PROTEIN DOMAINS

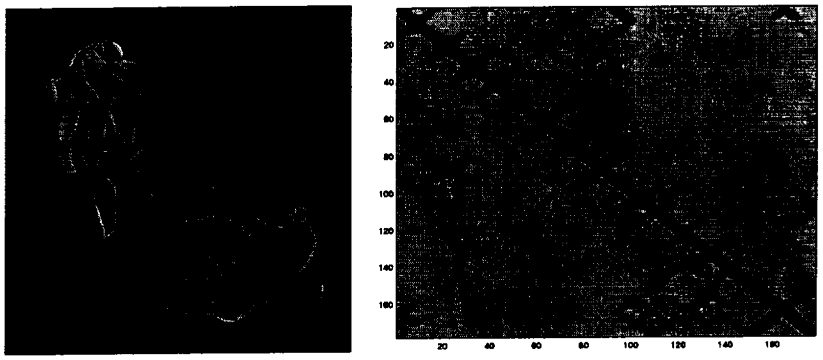
M. P. SAMANTA, S. LIANG
M/S 229-1, Nasa Ames Research Center,
Mountain View, CA, USA-94035
{msamanta,sliang}@nas.nasa.gov

H. ZHA
Department of Computer Science and Engineering,
Pennsylvania State University, PA, USA-16804
zha@cse.psu.edu

We present a novel computational method for automatic assignment of protein domains from structural data. At the core of our algorithm lies a recently proposed clustering technique that has been very successful for image-partitioning applications. This graph-theory based clustering method uses the notion of a normalized cut to partition an undirected graph into its strongly-connected components. Computer implementation of our method tested on the standard comparison set of proteins from the literature shows a high success rate (84%), better than most existing alternatives. In addition, several other features of our algorithm, such as reliance on few adjustable parameters, linear run-time with respect to the size of the protein and reduced complexity compared to other graph-theory based algorithms, would make it an attractive tool for structural biologists.

1 Introduction

Understanding the biological functions of proteins is one of the major challenges of the post genomic era [1,2,3,4,5]. The task would be vastly simplified, if the correlation between protein structures and sequences is properly understood, because the three-dimensional (3D) shapes of proteins may provide vital clues about their functions. An important lesson learned from research on protein structures is that the number of evolutionarily distinct proteins is finite. Families of evolutionarily related proteins share the same folding architecture. It is estimated that the total number of such folding architectures is of the order of thousands [1], which is much smaller than the total protein space. Some of these proteins can be further decomposed into domains, broadly defined as compact sub-structures in the 3D structure of a protein [6,7,8]. Some domains can carry out specific functions and also fold autonomously [10]. Due to such unique properties, domains are believed to serve as more fundamental units of evolution and in many ways more basic blocks of the protein universe [9]. Therefore, cataloging domains will enrich the database of protein domain fam-



(a) T-Cell Surface Glycoprotein (PDB code: 3CD4) contains two domains, as it is clearly visible in the figure.

(b) W matrix of protein 3CD4.

Figure 1: Molecule 3CD4 and its W matrix.

ilies and eventually help in protein homology detection. Moreover, recognizing the domain structures of a novel protein from its sequence will also improve structural prediction by threading. Because of such overwhelming importance of domains in structural classification as well as functional understanding of proteins, several databases (PFAM[11], CATH[12], SCOP[13], DALI[1,14]) maintain lists of domains of all the proteins of the Protein Data-bank (PDB)[15,16], which is the largest database of protein structures.

As the PDB database keeps growing exponentially, keeping the domain databases up-to-date turns out to be a challenging task. Initially, most of those databases were maintained manually with help from human experts. Lack of high-quality automated tools made the process slow and prone to errors. Although some automated algorithms have been proposed in recent years[1,7,8,18,27,28,30], they are still not very efficient[17]. After conducting an extensive comparison of all leading algorithms in 1998, Jones found that they worked correctly for only 65-75 percent of cases[17]. Moreover, the algorithms in her study all agreed only for 55 percent of the test-cases. Better techniques were proposed in the following years[7,18], but the problem of automated protein decomposition is far from solved.

Search for efficient domain decomposition techniques has been active since the 1970s[19,20,21,22,23,24], but the early efforts were inconclusive due to lack of analyzed protein structures. The methods tried in the early years included analysis of C^α-C^α distance-maps [20,21,22], minimum packing density of C^α atoms [19], comparing interface area between two chains [24], estimating maximum buried surface area [23], etc. Attempts were also made to predict domains solely from the sequence data [25,26]. The problem of domain decomposition from structural data received renewed attention in recent years due to exponential growth in the size of PDB over the last decade [15,16]. Sophisticated algorithms were suggested, some based on refinement of older ideas and some others using completely new concepts. These recent algorithms tried to obtain domains using inter-residue contacts [8], minimization of chain fragmentation [27], search for presence of hydrophobic cores [28,29], inter-domain dynamics [1,30], dendogram based on distance maps [27], multi-state partitioning using an Ising chain type model [18] and a graph-theory based network-flow algorithm [7].

In this paper, we use a powerful, normalized-cut based approach to partition proteins into domains. It is a graph-theory based method that has been quite successful for image-partitioning applications. In the biological field, Xing *et. al.* applied a similar technique for clustering DNA-micro-array data [34], but to the best of our knowledge, noone has applied it to the domain decomposition problem yet. A computer implementation of our algorithm, tried on standard test-set of 55 proteins from the literature [17], shows 84% rate of success, higher than most other existing algorithms [7,17]. Among the 30 single-domain proteins in the test-set, our method correctly identifies all but two. For the 20 two domain proteins, it achieves 80 percent rate of success. This success rate improves to 100 percent if the program knows beforehand that there are only two domains in the protein and then tries to find out the locations of these domains. For multi-domain proteins, the success rate is poorer but we also note that the test-set itself is very small. Only 5 out of 55 proteins in Jones' test-set had more than two domains. Therefore the results may be inconclusive in this case.

Our method has several other advantages. We find that even for proteins where our method gives incorrect results, the cut-value falls within a narrow range out of all possible values. This can be very helpful for semi-automatic identification of domains where human assistance can be taken only for proteins where the cut values fall within the narrow range. In addition, our method needs very few adjustable parameters and runs in linear time with respect to the size of protein. Also the implementation is cleaner than other graph-theory based domain-decomposition algorithms, because our method does not need to add unphysical source and sink nodes [7].

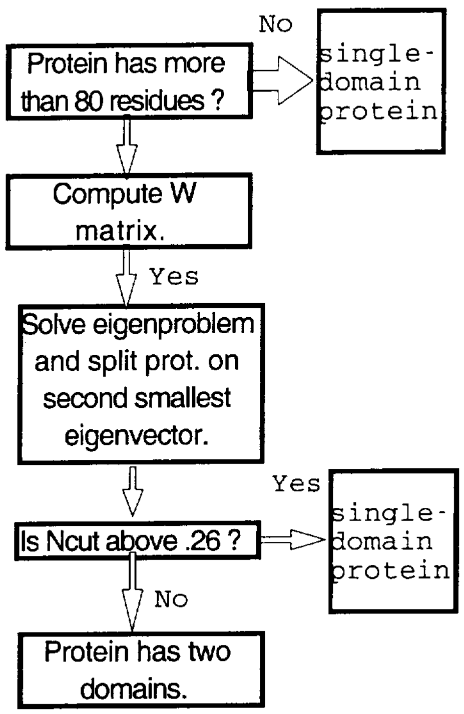


Figure 2: Flow Chart showing steps in the algorithm.

2 Methods

In this section, we explain our algorithm for obtaining domains of a protein from its structure (see Fig. [2]). For a given protein, we start with its structure downloaded from the PDB database and represent it as a weighted, undirected graph, where we consider the C_α atoms of the protein as vertices of the graph. Edge-weight w_{ij} between any pair of vertices (i, j) is computed using the following equation:

$$\begin{aligned} f_{ij} &= exp[-(\tfrac{d_{ij}-\bar{d}}{d_0})] + \beta_{ij} && (1) \\ w_{ij} &= f_{ij} \qquad \text{if} f_{ij} < 1 \\ &= 1 \qquad \text{if} f_{ij} \geq 1. && (2) \end{aligned}$$

In Eq. [2], d_{ij} is the distance between C^{α} atoms of residues i and j. d_0, $\bar{d}$ are constant parameters. β_{ij} takes only two values - constant positive parameter β or 0, depending on whether residues i and j belong to the same beta-sheet or not.

For the complete protein, edge-weights w_{ij} can be represented by a symmetric matrix W. Physically, elements of the matrix W represent the contacts between different residues of the protein. Larger values of w_{ij} close to 1 correspond to closely located residues i and j of the protein. On the other hand, a small value of w_{ij} implies that the residues i and j are far apart. The particular functional form for w_{ij} given by Eq. [2] turns out to be not important for our partitioning method. We can choose other functional forms as well. As long as the terms in W represent how strongly the amino acids are connected, with higher value for stronger connection, our partitioning algorithm would work. Therefore, the W matrix can be chosen to be as simple as 1(0) based on whether distance between C_{α} atoms of two residues are less than(greater than) a certain cutoff value [8] or based on some very sophisticated approach [7].

Once we represent the protein as a graph as described above, our problem of identification of domains translates into identifying the connected components of the graph that are weakly connected to each other. Efficient graph-theoretic algorithms exist to solve this problem. Among them, we choose the one that partitions the graph using the notion of a normalized cut. It is a clustering algorithm used routinely in the image-partitioning field [33]. In the graph-theoretic language, our problem can be stated as follows: we seek to partition the vertices V of a weighted, undirected graph $G(V, W)$ into two disjoint sets V_1 and V_2, where the contact is large between the vertices of the same set and small between vertices of different sets. Intuitively, a good partition (V_1, V_2) of the graph (V) should minimize the sum of the weights for edges connecting two sub groups. Mathematically, we need to minimize the sum

$$cut(V_1, V_2) = \sum_{i \in V_1} \sum_{j \in V_2} w_{ij} \tag{3}$$

to get the best partition. Unfortunately, such an approach is prone to result in unbalanced partitions. We can check easily that a partition with only one vertex in one subgroup (V_1) and the rest of the vertices in other subgroup (V_2) may give very low sum in Eq. [3], but that is not the optimum solution that we are looking for. Therefore the function in Eq. [3] needs to be normalized to get the correct answer.

There are many possible ways to normalize the function in Eq. [3]. Shi and Malik studied many such alternatives [33] and found that minimization of the function

$$Ncut(V_1, V_2) = \frac{cut(V_1, V_2)}{assoc(V_1, V)} + \frac{cut(V_1, V_2)}{assoc(V_2, V)}, \tag{4}$$

gives the optimum partition of the graph. In Eq. [4] $assoc(V_m, V)$ is defined as

$$assoc(V_m, V) = \sum_{i \in V_m} \sum_{j \in V} w_{ij}. \tag{5}$$

It can be shown that $Ncut$ has a maximum range of $(0, 2)$.

In order to find out the best partition of the vertices of the graph, one obvious approach would be to go over all possible partitions of the set V, compute $Ncut$ for each partition using Eq. [4] and then pick the partition that gives lowest $Ncut$. However, such an approach is NP-complete [33] and therefore not practically feasible for large proteins. However, it is possible to get an approximate solution for the partitioning problem in the following way. We consider the the generalized eigenvalue problem:

$$(D - W)v = \lambda D v, \tag{6}$$

where W is the edge-weight matrix of the graph and D is a diagonal matrix with each element as a sum of the rows of W. After solving the eigenvalue problem, we obtain the eigenvector corresponding to the second smallest eigenvalue of Eq. [6]. It is proved[33] that an approximate solution for the optimum partition of the graph is given by considering the positive and negative elements of the chosen eigenvector as two subsets. Moreover, the eigenvector for the second smallest eigenvalue of Eq. [6] can be computed in O(N) time making such an approach computationally very efficient.

The mathematical derivation of the eigenvector-based procedure is given in Ref. [33]. Instead of reproducing it here, we attempt to physically justify it by showing analogy to another problem. In quantum mechanics, we describe the eigenstates of the hydrogen atom in terms of s, p and d orbitals, all of which have different shapes. The s-state is spherical with no nodes, and the p-state is dumbbell-shaped with one node. On the other hand, the s-state is the lowest energy state and the p-state state has the second smallest eigen-energy. Since energy is an eigenvalue of Schrodinger's equation, we see that the eigenvector corresponding to the second smallest eigenvalue attempts to partition space in two distinct lobes. Our procedure here is analogous.

Once we identify the optimum partition of the protein using the above-mentioned procedure, we need to decide whether to accept the cut or reject it. There are many single-domained proteins which need not be split into two domains. The decision process is conducted by comparing the obtained smallest *Ncut* value with a predetermined parameter *cutoff*. If *Ncut* is higher than *cutoff*, the cut is rejected and the protein is concluded to be single-domained. On the other hand, if *Ncut* is lower than *cutoff*, the cut is accepted. Each segment is considered to be a domain of the protein and subsequently checked for the possibility of further subdivisions by reapplying the algorithm on the individual segments.

The algorithm described above is generally applicable to any partitioning problem. For the specific case of protein domain-decomposition, some additional modifications need to be made. Following Richardson's broad guidelines[6] reflecting a commonly accepted viewpoint on the definition of domains, we do not accept protein segments less than 40 residues long as domains and therefore do not attempt to cut proteins less than 80 residues long. Moreover, to avoid fragmentation, if our algorithm predicts parts of domains which are less than 20 residues long, we insert such short segments back into the other part of the chain as a post-processing step.

Additional steps are taken to make the algorithm numerically efficient. Firstly, we truncate small terms of W to zero. When the distance between two residues is above $25\mathring{A}$, w_{ij} is truncated to zero. This gives us a sparse W matrix improving speed at no performance sacrifice. For such sparse W, the eigenvalues and eigenvectors in Eq. [6] can be computed in $O(N)$ steps using the Lanczos algorithm, an approximate, recursive method. In comparison, the normal time for calculation of eigenvalues is $O(N^3)$. Moreover, to improve the approximate partitioning algorithm, we partition the graph not solely on values of the eigenvector above and below zero, but also take few more cutoff points near zero to see whether the *Ncut* value improves. Such a procedure is originally suggested in Ref. [33] and we find it to improve the quality of our results.

3 Estimation of Model Parameters

Four important parameters ($\bar{d}$, d_0 and β and *cutoff*) need to be estimated for optimum performance of our algorithm. For calibration of these parameters, we choose a method similar to Ref. [7]. We consider a calibration set of 206 proteins[35] for which domains are already known and available in the literature[8]. 47 of them are two-domained proteins the remaining 159 are single-domained. We apply our algorithm to all these proteins with different values of parameters $\bar{d}$, d_0, β and *cutoff* and compare results with known solutions. The values of the parameters for which the domains of most proteins are correctly identified are chosen as standard parameters. Based on this analysis, we found that

$\bar{d} = 10.33$, $d_0 = 2$, $\beta = .01$ and *cutoff*=.26 give the best results.

With the chosen form of the W matrix and associated parameters, we discuss here the implications of varying the parameters. We find that if $\bar{d}$ is changed to larger values, w_{ij} becomes 1 for many residue-pairs. Therefore, the resolution of the method goes down and *Ncut* gets larger. Resolution improves for smaller $\bar{d}$. However, if $\bar{d}$ is too small, all off-diagonal elements become very small. So that is not acceptable either. Some middle-value of $\bar{d}$ is more desirable, and the method optimizes for $\bar{d} = 10.33\mathring{A}$. On the other hand, d_0 is a scaling parameter. It effectively scales up the high-resolution numbers that were obtained with low $\bar{d}$.

4 Results and Discussion

We illustrate the general procedure discussed in section [2] with the example of T-Cell Surface Glycoprotein (PDB code: 3CD4). This protein has two domains clearly identifiable in Fig. [1(a)]. We represent the protein as a graph and compute the adjacency matrix W. The W matrix is shown in Fig. [1] as an image plot. Based on the eigenvector corresponding to the second smallest eigenvalue in Eq. [6], we partition the protein into two groups of residues (1,97) and (98,178). The value of *Ncut* for this partition is 0.17 which is much smaller than *cutoff* parameter 0.26. Therefore the cut is accepted and two segments are chosen as two domains of the protein. Attempts at further partitioning of the individual segments produce *Ncut* values larger than .26 and therefore such subdivisions are rejected. Hence, our algorithm gives two domains for the protein, closely matching expert opinion from the literature [(1,98) and (99,178)].

To compare the overall quality of our method with other methods in the literature, we try it on the standard set of 55 proteins from the literature [17]. The set of protein includes 30 single-domain proteins [a], 20 two-domain proteins [b], two three-domain proteins [c] and three four-domain proteins [d]. Domains for all these proteins are available from the literature [8]. Structures of some of the proteins in the PDB database have been corrected and saved under new names in PDB since the original paper came out in 1998. Therefore, we updated

[a]one-domain: 2aak, 2ace, 1bbhA, 1bbpA, 1brd, 1fxiA, 1gky, 2gmfA, 1gmpA, 1gox, 1ofv, 1pyp, 1rbp, 1rcb, 1rveA, 1snc, 1tie, 1tlk, 1ula, 1bksA, 2azaA, 2ccyA, 2rn2, 2stv, 2tmvA, 3chy, 3cla, 3dfrA, 4blmA, 5p21

[b]two-domain: 1ezm, 1fnb, 1gpb, 1lap, 1pfkA, 1ppn, 1rhd, 1sgt, 1vsg, 1bksB, 2cyp, 2had, 3cd4, 1g6nA, 3pgk, 4gcr, 5fbp, 8adh, 8atcA, 8atcB

[c]three-domain: 1phh, 3grs

[d]four-domain: 1atnA, 3pgmA, 8acn

Table 1: Domain decomposition of 25 multi-domain proteins of the standard comparison set are shown here. Column 'Expert Opinion in Lit.' shows the domains identified by experts in the literature. Column 'Normalized Cut' shows results from our algorithm. For the remaining 30 proteins which are single-domained, three proteins 2ACE and 2TMVP are identified incorrectly as two-domained by our algorithm.

Protein	Expert Opinion in Lit.	Normalized Cut	Accuracy
2 domains:			
1EZM	(1-134), (135-298)	(1-146,170-197), (147-169,198-298)	87%
1FNB	(19-161), (162-314)	(19-153), (154-314)	98%
1GPB	(19-489), (490-841)	3 domains	wrong
1LAP	(1-150), (171-484)	(1-158), (159-484)	100%
1PFKA	(0-138,251-301), (139-250,302-319)	(0-142,255-319), (143-251)	98%
1PPN	(1-10,112-208), (21-111,209-212)	(1-212)	wrong
1RHD	(1-158), (159-293)	(1-157), (158-293)	100%
1SGT	(22-123, 234-245), (129-233)	(16-245)	wrong
1VSGA	(1-29, 92-251), (42-75, 266-362)	(1-33,86-255),(34-85,256-362)	100%
1BKSB	(9-52, 86-204), (53-85, 205-393)	(3-53,87-204), (54-86,205-394)	100%
2CYP	(3-145, 266-294), (164-265)	(2-144, 266-294), (145-265)	99%
2HAD	(1-155, 233-310), (156-229)	(1-310)	wrong
3CD4	(1-98), (99-178)	(1-97), (98-178)	100%
1G6NA	(1-129), (139-208)	(7-128), (129-204)	100%
3PGK	(1-185, 403-415), (200-392)	(1-195,390-415), (196-389)	100%
4GCR	(1-83), (84-174)	(1-81), (82-173)	99%
5FBP	(6-201), (202-335)	(6-199), (200,334)	100%
8ADH	(1-175, 319-374), (176-318)	(1-177, 318-374), (178-317)	99%
8ATCA	(1-137, 288-310), (144-283)	(1-140, 289-309), (141-288)	100%
8ATCB	(8-97), (101-152)	(8-98), (99-153)	100%
3 domains:			
1PHH	(1-155), (176-290), (291-394)	(1-69,100-180,269-341) (70-99,181-268,342-394)	wrong
3GRS	(18-157, 294-364), (158-293), (365-478)	(18-60,108-159,291-364), (61-107,160-220,242-290), (221-241,365-478)	87%
4 domains:			
1ATNA	(1-32, 70-144, 338-372), (33-69), (145-180, 270-337), (181-269)	(1-33,69-136,339-372), (34-68,186-257), (137-185,258-338)	95%
3PMGA	(1-188), (192-315), (325-403), (408-561)	(1-193), (194-419), (420-561)	wrong
8ACN	(2-200), (201-317), (320-513), (538-754)	(2-67,151-208,231-312), (68-150,510-530), (209-230,313-508), (531-754)	wrong

proteins 1aak by 2aak, 1ace by 2ace, 1gmfA by 2gmfA, 1wsy by 1bks, 1fnr by 1fnb, 2pmg by 3pmg and 3gap by 1g6n to reflect this change.

The results of our comparison are shown in table[1]. Using the definition of correctness of Islam[8,7], our method gives 84 percent accuracy. We have nine incorrect results - proteins with PDB codes 2ace, 2tmvP, 1gpb, 1ppn, 1sgt, 2had, 1phh, 3pmgA and 8acn.

By analyzing the *Ncut* numbers for the proteins, for which our algorithm failed we note that they resulted in *Ncut* values with a much smaller subrange out of maximum range of $(0-2)$. This allows us to suggest a better method for protein domain decomposition. We can consider all the proteins and then for the ones where the cut value is outside the subrange, we can surely say that the cut can be accepted. On the other hand, for the proteins where the cut-value is within a small subrange 'gray-area', we may discuss with an expert to accept or reject the cut. The whole range $(0,2)$ of minimum *Ncut* values can be divided into (i) sure cut $(0-.20)$, gray region $(.20-.30)$ and sure not a domain $(.31-2)$. This will greatly reduce the burden on human experts maintaining the PDB database, because they can concentrate on a smaller group of proteins rather than all the proteins. We note that using this strategy, we can correctly identify all the proteins in Jones list except three. These three are proteins with PDB codes 1sgt, 2had and 1atn, some which have history of difficulty of identification even by human experts[18].

5 Conclusion

In this paper, we use a powerful, normalized-cut based approach to partition proteins into domains. This graph-theory based approach, borrowed from the image-partitioning field, shows a good rate of success when applied to the protein domain decomposition problem. A computer implementation of our algorithm, tried on a commonly used standard test-set of 55 proteins[17], obtains 84 percent rate of success, higher than most other existing algorithms[17,7]. Also this method needs very few adjustable parameters and runs in linear time with respect to the size of protein. Moreover, we find that even for proteins where our method gives incorrect results, the cut-value falls within a narrow range out of all possible values. Therefore, our method will be useful for automatic domain databases as well as semi-automatic ones where the aid of a human expert is taken for difficult proteins.

Acknowledgements

Manoj Samanta's work is supported by NASA contract DTTS59-99-D-0437/A61812D to CSC. Shoudan Liang is supported by NASA Ames DDF and contract 755-30-11.

1. L. Holm and C. Sander *Science* **273**, 595-602 (1996).
2. M. Gerstein and M. Levitt *Proc. Nat. Acad. Sci. USA* **94**, 11911-11916 (1997).
3. M. Gerstein *Nature Struct. Biol.* **11**, 960-963 (2000).
4. M. Turcotte, S. H. Muggleton and M. J. E. Sternberg *J. Mol. Biol.* **306**, 591-605 (2001).
5. P. Aloy, E. Querol, F. X. Aviles, and M. J. E. Sternberg *J. Mol. Biol.* **311**, 395-408 (2001).
6. J. S. Richardson *Adv. Protein Chem.* **34**, 167-339 (1981).
7. Y. Xu, D. Xu, and H. N. Gabow *Bioinformatics* **16**, 1091-1104 (2000).
8. S. A. Islam, J. Luo and M. J. E. Sternberg *Protein Engg.* **8**, 513-525 (1995).
9. S. Das and T. F. Smith *Adv. Prot. Chem.* **54**, 159-183 (2000).
10. Z. Y. Peng and L. C. Wu *Adv. Prot. Chem.* **53**, 1-47 (2000).
11. A. Bateman, E. Birney, L. Cerruti, R. Durbin, L. Etwiller, S.R. Eddy, S. Griffiths-Jones, K. L. Howe, M. Marshall, and E. L. Sonnhammer *Nucleic Acids Research* **30**, 276-280 (2002).
12. C. A. Orengo, A. D, Michie, S. Jones, D. T. Jones, M. B. Swindells and J. M. Thornton *Structure* **5**, 1093-1108 (1997).
13. A. G. Murzin, S. E. Brenner, T. Hubbard and C. Chothia *J. Mol. Biol.* **247**, 536-540 (1995).
14. L. Holm and C. Sander *Nucleic Acids Research* **26**, 316-319 (1998).
15. H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov and P. E. Bourne *Nucleic Acids Research* **28**, 235-242 (2000).
16. F. C. Bernstein, T. F. Koetzle, G. J. B. Williams, B. J. Meyer, M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi and M. Tasumi *J. Mol. Biol.* **112**, 535-542 (1977).
17. S. Jones, M. Stewart, A. Mitchie, M. B. Swindells, C. Orengo, and J. M. Thornton, *Protein Sci..* **7**, 233-242 (1998).
18. W. R. Taylor *Protein Engg.* **12**, 203-216 (1999).
19. G. D. Rose *J. Mol. Biol.* **134**, 447-470 (1979).
20. G. M. Crippen *J. Mol. Biol.* **233**, 123-138 (1978).
21. K. Nishikawa, T. Ooi, T. Isogai and N. Saito *J. Phys. Soc. Jpn.* **32**, 1331-1337 (1972).

22. A. Lijas and M. G. Rossman *Ann. Rev. Biochem.* **43**, 475-507 (1974).
23. A. Rashin *Nature (London)* **291**, 85-87 (1981).
24. S. J. Wodak and J. Janin *Biochemistry* **20**, 6544-6552 (1981).
25. B. Busetta and Y. Barrans (1984). *Biochim Biophys Acta* **790**, 117-124.
26. T. Kikuchi G. Némethy and H. A. Scheraga *J. Protein Chem* **7**, 427-471 (1988).
27. R. Sowdhamini and T. L. Blundell *Protein Science* **4**, 506-520 (1995).
28. M. B. Swindells *Protein Sci.* **4**, 103-112 (2001).
29. M. B. Swindells *Protein Sci.* **4**, 93-102 (1995).
30. L. Holm and C. Sander *Proteins Struct. Funct. Genet.* **12**, 256-268 (1994).
31. T. H. Cormen, C. E. Leiserson and R. L. Rivest, Introduction to Algorithms, 2nd ed. *MIT Electrical Engineering and Computer Science* (2001).
32. L. R. Ford and D. R. Fullerson *Flows in Networks* Princeton University Press, Princeton, New Jersey (1962).
33. J. Shi and J. Malik *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**, 888-905 (2000).
34. E. P. Xing and R. M. Karp *Proceedings of The Ninth International Conference on Intelligence Systems for Molecular Biology* (2001).
35. We start with 284 proteins for which domains are known as summarized in Ref. [8]. From this list we keep aside 55 proteins that belong to the test-set. We also exclude multidomain proteins, proteins without correct structures (1bop, 1ctc, 1prf) and proteins smaller than 40 residues (1bbo, 1cpcA, 1cpcL, 4rcrH).